IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Kay-Yut Chen	§	Art Unit:	3623
		§		
Serial No.:	09/944,969	§		
		§	Examiner:	Linda Mary Krisciunas
Filed:	August 30, 2001	§		
		§		
For:	Method and Apparatus for Modeling Business Processes	§	Atty. Dkt. No.:	10004567-1 (HPC.0328US)

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION UNDER 37 C.F.R. § 1.131

Dear Sir:

I, Kay-Yut Chen, state as follows:

1. I am the inventor of the above-referenced application.

2. The document attached as Exhibit A is a true and correct copy of the Invention Disclosure pertaining to subject matter described and claimed in the above-referenced patent application, which I submitted to the legal department of the Hewlett-Packard Company prior to March 8, 2001.

3. The page of Exhibit A bearing the number "3" at its bottom describes a "Business Model Definition Process," which provides a framework having several listed elements, including: (a) decision making entities; (b) possible decision space; (c) decision making process tree; (d) information set; (e) outcome function; and (f) payoff function.

4. The page of Exhibit A bearing the number "2" at its bottom shows a figure that is substantially the same as Figure 1 of the present application.

5. The page of Exhibit A bearing the number "4" at its bottom describes a "Script Translation Process" (in section 3). The Script Translation Process translates the output of the Business Model Definition Process and a Business Model Calibration Process into a software representation that is stored as scripts.

Date of Deposit: May 30, 2006

I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as **first class mail** with sufficient postage on the date indicated above and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313.

Ginger Yount

Ginger Yount

6. The page of Exhibit A bearing the number "4" at its bottom, in section 4, also describes taking the generated scripts and providing a simulation that resembles the original business environment, where the simulation can be interactive with players that are human beings.

7. The page of Exhibit A bearing the number "1" in the section titled "Problems Solved by Invention," states that a user can define different scenarios under which the simulation can be conducted to provide different comparisons and evaluations.

8. The above establishes conception of the present invention prior to March 8, 2001.

9. Moreover, the subject matter of the Invention Disclosure (Exhibit A) referred to above was implemented in software prior to March 8, 2001. The attached Exhibit B lists various software modules that implement the MUMS system, which includes the subject matter of the Invention Disclosure referred to above.

10. The software modules (listed in Exhibit B) of the MUMS system were successfully executed in a computer prior to March 8, 2001. I determined at that time that the MUMS system worked for its intended purpose.

11. Exhibit C contains the title page and various pages of a document titled "Script Users Guide," which was dated prior to March 8, 2001. The Script Users Guide describes elements of a script used in the MUMS system.

12. Exhibit D is an Installation and Maintenance Guide dated prior to March 8, 2001 that describes installation and maintenance operations for the MUMS software. The Installation and Maintenance Guide was used to install the MUMS software modules in a computer, which software modules were successfully executed prior to March 8, 2001.

13. The first page of the Invention Disclosure (Exhibit A) indicates that results were obtained from successful use of the invention prior to March 8, 2001.

14. Exhibit E is a paper by Gary Charness and Kay-Yut Chen, entitled "Minimum Advertised-Price Policy Rules and Retailer Behavior: An Experiment by Hewlett-Packard" (2002). In the section starting on page 65 of Exhibit E, titled "Experimental Procedure," reference is made to execution of the software that makes up the MUMS system which contained an implementation of the invention. The MUMS system worked for its intended purpose.

15. The above facts establish completion of my invention (both conception and actual reduction to practice) prior to March 8, 2001.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: 5/25/2006

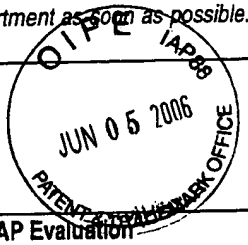
KAY-YUT CHEN

HEWLETT PACKARD **INVENTION D CLOSURE**

PAGE ONE OF ____

PDNO 10004567 DATE RCVD ATTORNEY TXL/STL

Instructions: *The information contained in this document is* ***COMPANY CONFIDENTIAL*** *and may not be disclosed to others without prior authorization. Submit this disclosure to the HP Legal Department as soon as possible. No patent protection is possible until a patent application is authorized, prepared, and submitted to the Government.*



Descriptive Title of Invention:

Business Processes Modeling Module

Name of Project: CPG Return Policy Evaluation/IPG UMAP Evaluation

Product Name or Number:
none

Was a description of the invention published, or are you planning to publish? If so, the date(s) and publication(s):

no

Was a product including the invention announced, offered for sale, sold, or is such activity proposed? If so, the date(s) and location(s):

no

Was the invention disclosed to anyone outside of HP, or will such disclosure occur? If so, the date(s) and name(s):

If any of the above situations will occur within 3 months, call your IP attorney or the Legal Department now at 1-898-4919 or 970-898-4919.

Was the invention described in a lab book or other record? If so, please identify (lab book #, etc.)

An executive summary of the CPG Return Policy Evaluation is available on HPL/STL/DTD website. A paper about the IPG UMAP Evaluation project (which referred to the results of using the invention but not disclosing the invention) is submitted to Inteface

Was the invention built or tested? If so, the date:

Invention was used in the above mentioned 2 projects in

Was this invention made under a government contract? If so, the agency and contract number:

No

Description of Invention: *Please preserve all records of the invention and attach additional pages for the following. Each additional page should be signed and dated by the inventor(s) and witness(es).*

A. Prior solutions and their disadvantages (if available, attach copies of product literature, technical articles, patents, etc.).
B. Problems solved by the invention.
C. Advantages of the invention over what has been done before.
D. Description of the construction and operation of the invention (include appropriate schematic, block, & timing diagrams; drawings; samples; graphs; flowcharts; computer listings; test results; etc.)

Signature of Inventor(s): Pursuant to my (our) employment agreement, I (we) submit this disclosure on this date: [ ].

Employee No.	Name	Signature	Telnet	Mailstop	Entity & Lab Name
461341	Kay-Yut Chen		857-7091	1U-2	HPL/STL/DTD

(If more than four inventors, include additional information on another copy of this form and attach to this document)

Form 3.1 IDF.DOC Rev. 10/25/99

HEWLETT PACKARD **INVENTION D..CLOSURE** COMPANY CON. .DENTIAL PAGE ____ OF ____

Signature of Witness(es): *(Please try to obtain the signature of the person(s) to whom invention was first disclosed.)*

The invention was first explained to, and understood by, me (us) on this date: [ ]

Full Name	Signature	Date of Signature
Full Name	Signature	Date of Signature

Inventor & Home Address Information: *(If more than four inventors, include addl. information on a copy of this form & attach to this document)*

Inventor's Full Name
Kay-Yut CHen

Street
610, Arcadia Terrace, #201

City: Sunnyvale | State: Ca | Zip: 94086

Do you have a Residential P.O. Address? P.O. BOX | City | State | Zip

Greeted as *(nickname, middle name, etc.)* | Citizenship: Hong Kong

Inventor's Full Name

Street

City | State | Zip

Do you have a Residential P.O. Address? P.O. BOX | City | State | Zip

Greeted as *(nickname, middle name, etc.)* | Citizenship

Inventor's Full Name

Street

City | State | Zip

Do you have a Residential P.O. Address? P.O. BOX | City | State | Zip

Greeted as *(nickname, middle name, etc.)* | Citizenship

Inventor's Full Name

Street

City | State | Zip

Do you have a Residential P.O. Address? P.O. BOX | City | State | Zip

Greeted as *(nickname, middle name, etc.)* | Citizenship

B) Problem Solved by Invention

The Business Process Modeler (BPM) translates a business process or a business model into a software simulation. This simulation can be interactive with human subjects. Furthermore, a user can define different scenarios under which the simulation can be conducted.

The BPM can be used to generate data under different business model and hence provides scientific evaluations and comparisons of different business models. These comparisons and evaluations are then used to make informed decisions of business variables.

Actual applications of this invention include but are not limited to the MAP policy design and evaluation project and the return policy design and evaluation project.

A) Prior Solutions and Their Disadvantages

To our knowledge, there is no solution to model general business processes as a software simulation. The closest prior solutions will be:

- Experimental economics software platforms employed and development by various academic institutions. Most of these platforms are problem specific with limited flexibility to simulate types of business processes and models that are not pre-coded into the system. The following are several disadvantages of these systems:
    - Limited flexibility. For example, a market-based system (such as MUDA developed at Caltech) cannot simulate return policies (a common business process employed between manufacturers and retailers).
    - Difficult to adapt to new situations. If a certain process characteristic is changed in the system, extensive programming is needed to effect the change. For example, if a double auction based market is changed to a call market, then recoding of the system is needed.

- A prototype of MUMS system that was developed as a joint research project between Caltech and HP Labs. The following are the major disadvantages of this system:
    - No support for real time processes. Only discrete time process is supported
    - Implementation of scripting language is incomplete. Many scenarios cannot be fully simulated by the script language and a code change is needed.
    - Based on obsolete networking and interfacing technologies (non TCP/IP or Window based)

C) Advantages of the Invention Over What Has Been Done Before

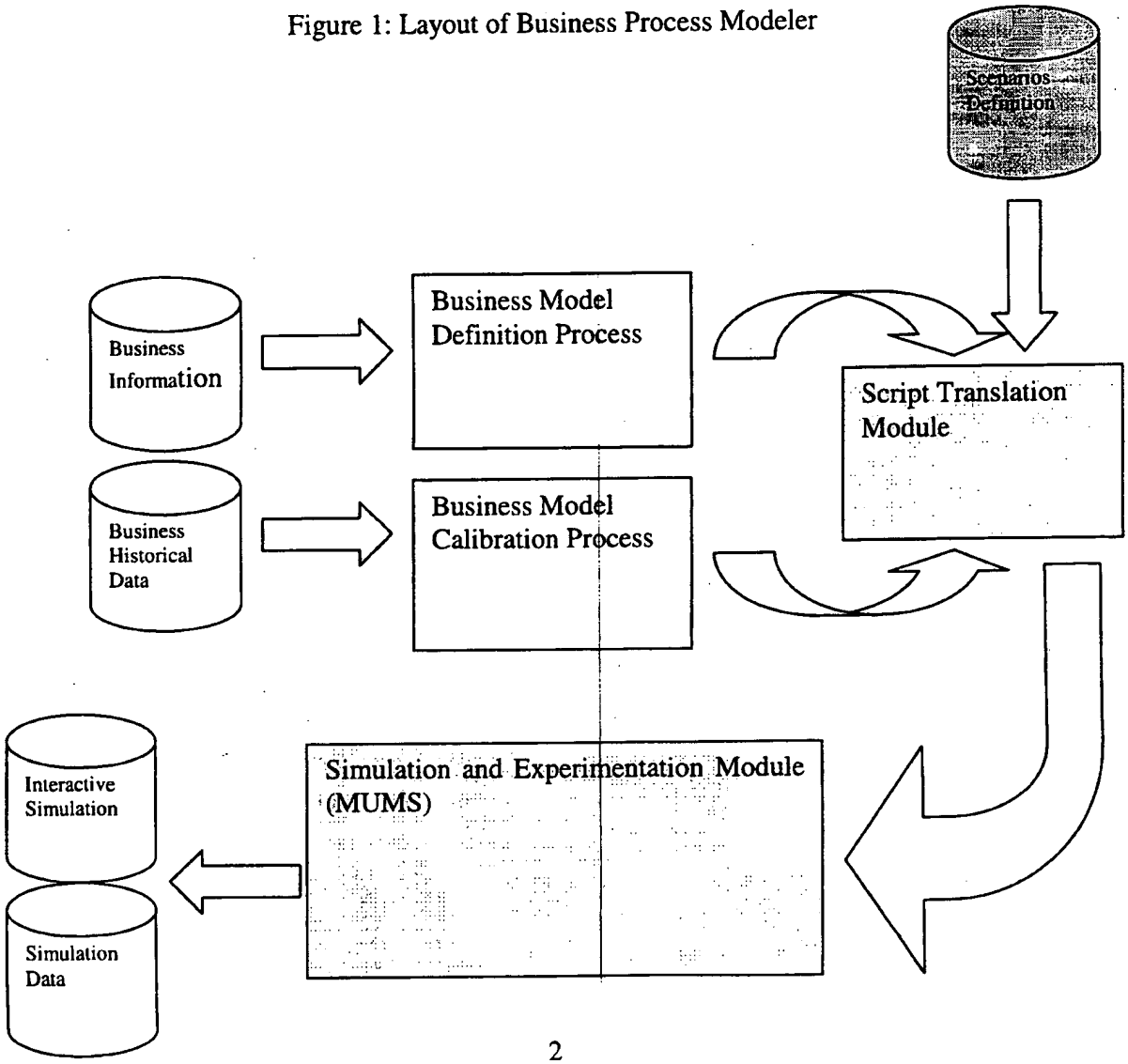
There are no other inventions that can prove general business model simulation with human interactions. The closest, the prototype developed at Caltech is incomplete and

cannot be used for many situations without redesign of the software. Our system is complete, flexible and supports the following features:

- Support discrete and continuous time business processes
- Database support to record data
- Client interface are window based and user-friendly
  - Support graphs
  - Spreadsheet
- Versatile scripting language including
  - Definition of functions
  - Use of local variables
  - Flow controls

D) Description of Construction and Operation of the Invention Construction/Functionality

Figure 1: Layout of Business Process Modeler



There are four major modules in the Business Process Modeler:

1) Business Model Definition Process
2) Business Model Calibration Process
3) Script Translation Process
4) Simulation and Experimentation Module

1) Business Model Definition Process

In this module, a business environment is translated into a representation of a pre-defined framework developed by HPL. This framework maps the business environment into the following elements:

a) Decision Making Entities

These are the decision makers in the environments. Examples are retailers in a reseller business model.

b) Possible Decision Space

These are decisions the decisions makers need to make. Examples are prices and stocking decisions in a reseller business model.

c) Decision Making Process Tree

This is a representation of the complete flow of the decision making process, primarily timing of the events.

d) Information Set

This defines the information that each decision-making entity has access to at each node of the Decision Making Process Tree.

e) Outcome function

This is the mathematical function that determines the outcome as a function of the decision variables.

f) Payoff function

This is the mathematical function that provides each decision-making entities with a "score". Examples are profit, return on investment and revenue growth in a reseller business model.

2) Business Model Calibration Process

The Business Model Calibration Process takes historical data as input. It analyzes historical data and provides estimations of the qualitative aspects of the outcome function and the payoff function (specified in the previous section).

3) Script Translation Process

The Script Translation Process translates the output of the Business Model Definition Process and the Business Model Calibration Process into a software representation developed by HPL.

Then variations of the software representation are generated based on pre-defined scenarios of that the original business environment.

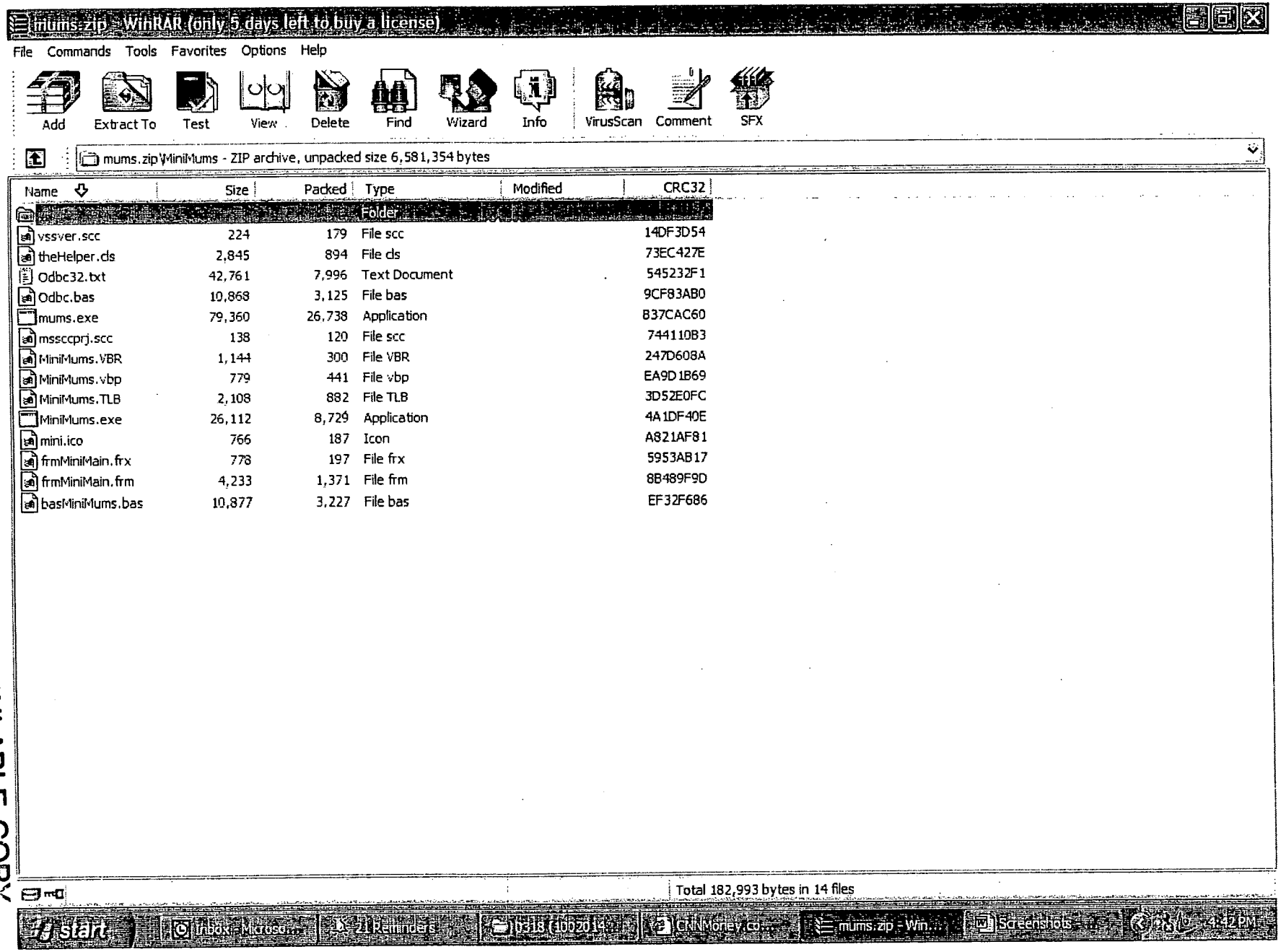
These software representations are store as computer files that are called "scripts".

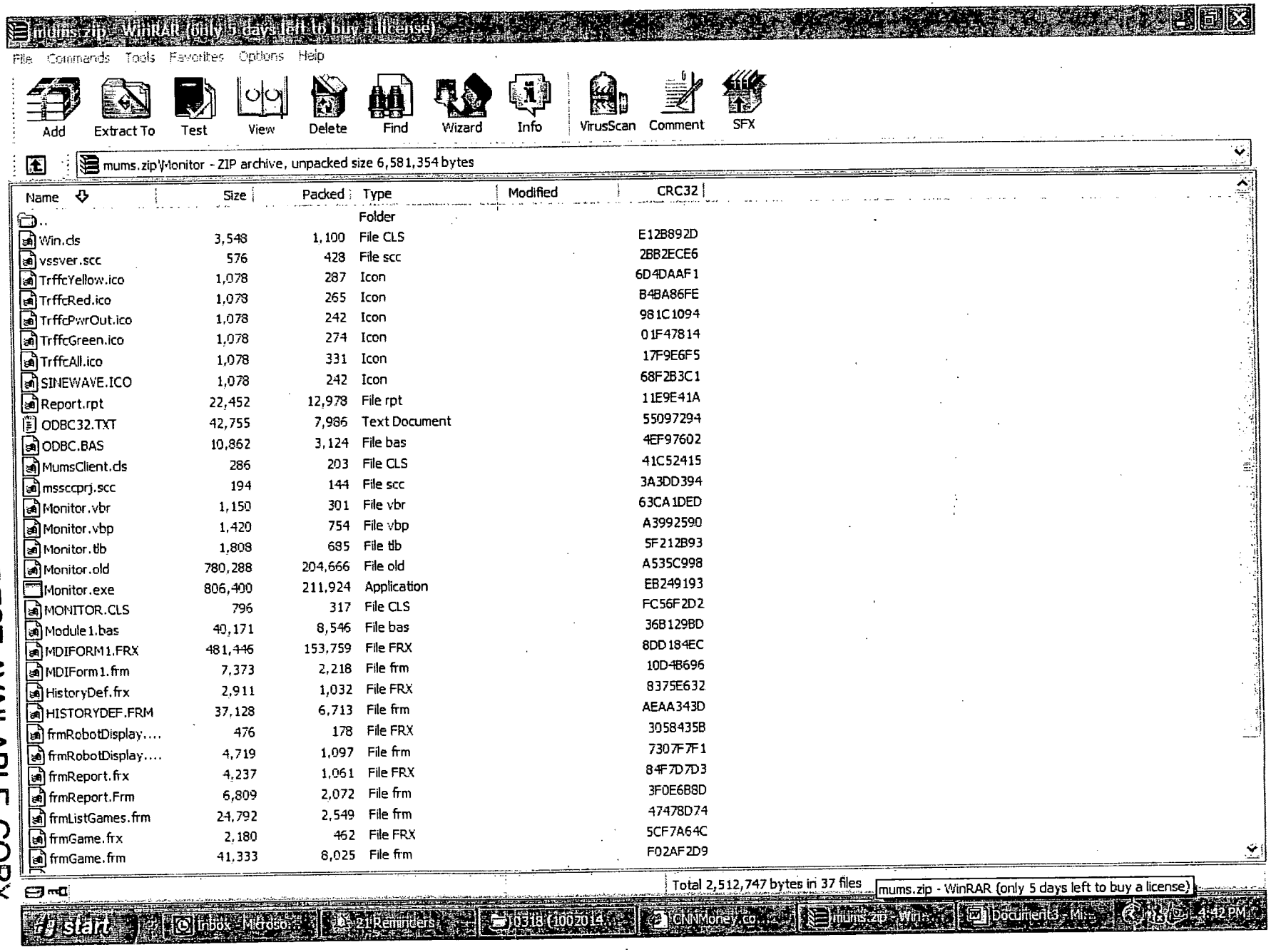
4) Simulation and Experimentation Module (MUMS)

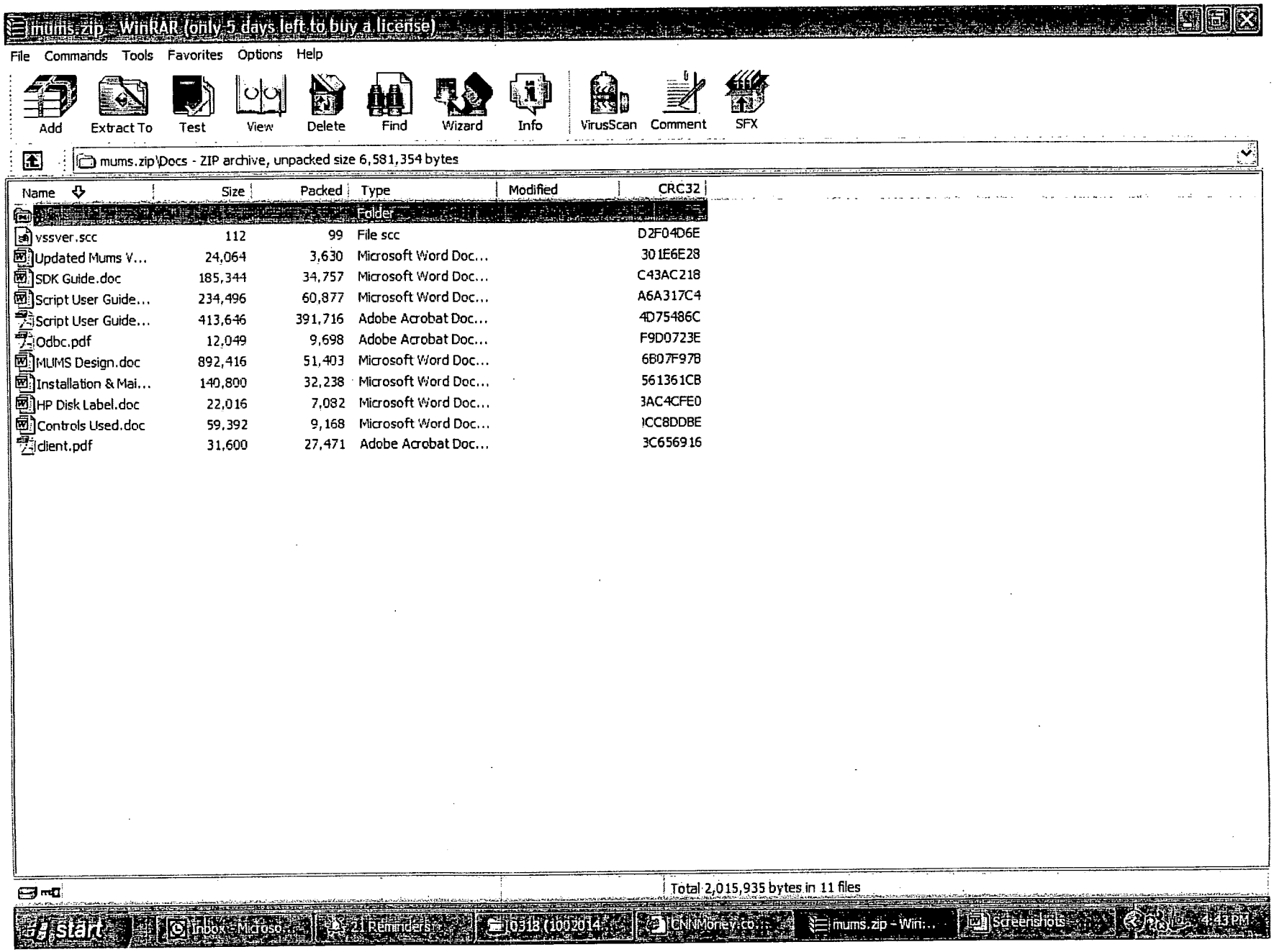
This module is also known as the MUMS (multi-user multi-stage) software system. It takes the scripts generated by the Script Translation Process and provides a simulation that resembled the original business environment, which provides the inputs of business information and data.

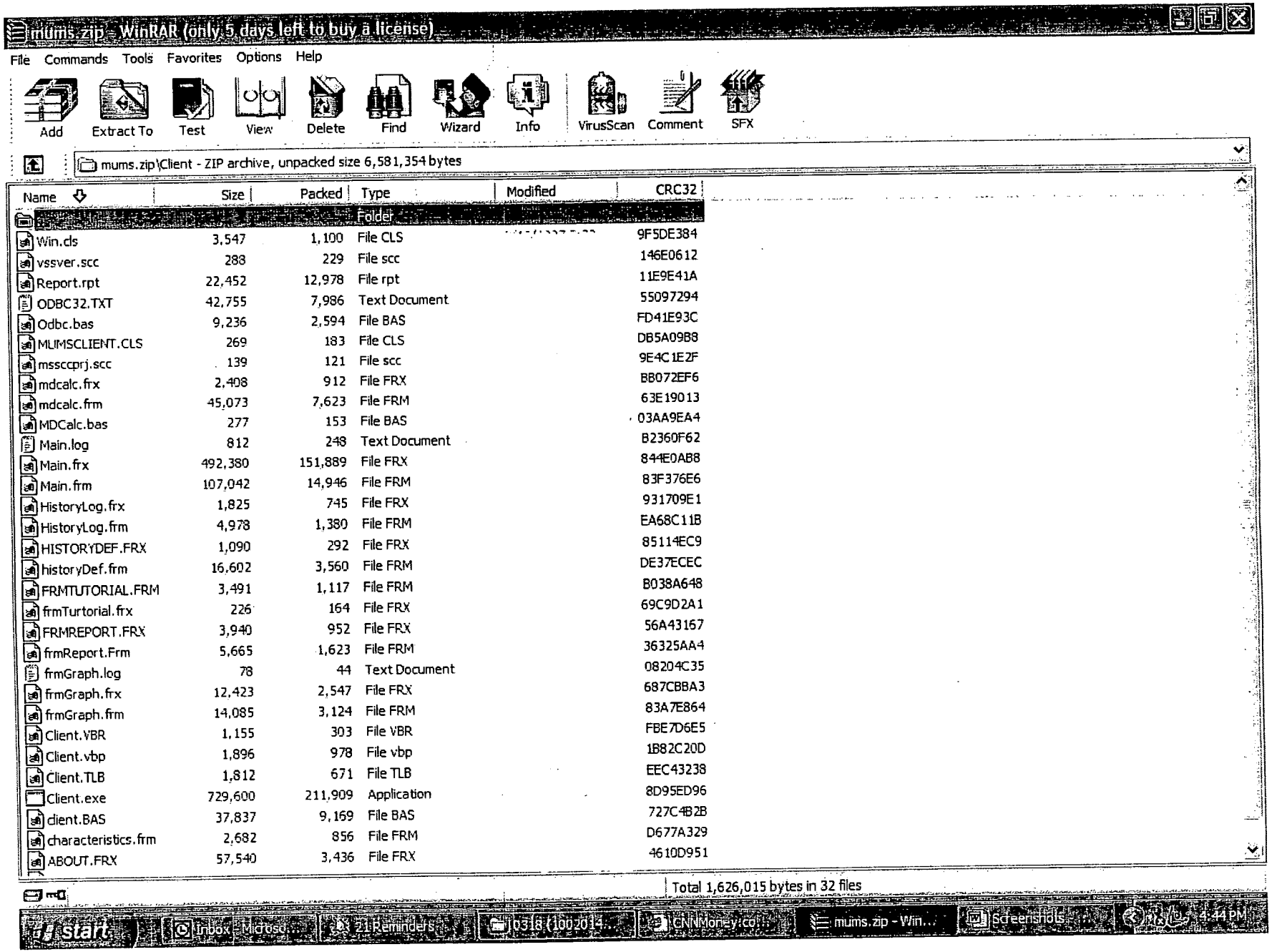
The simulation can be interactive as human beings. In particular, each decision-making entity can be "played" by a human subject.

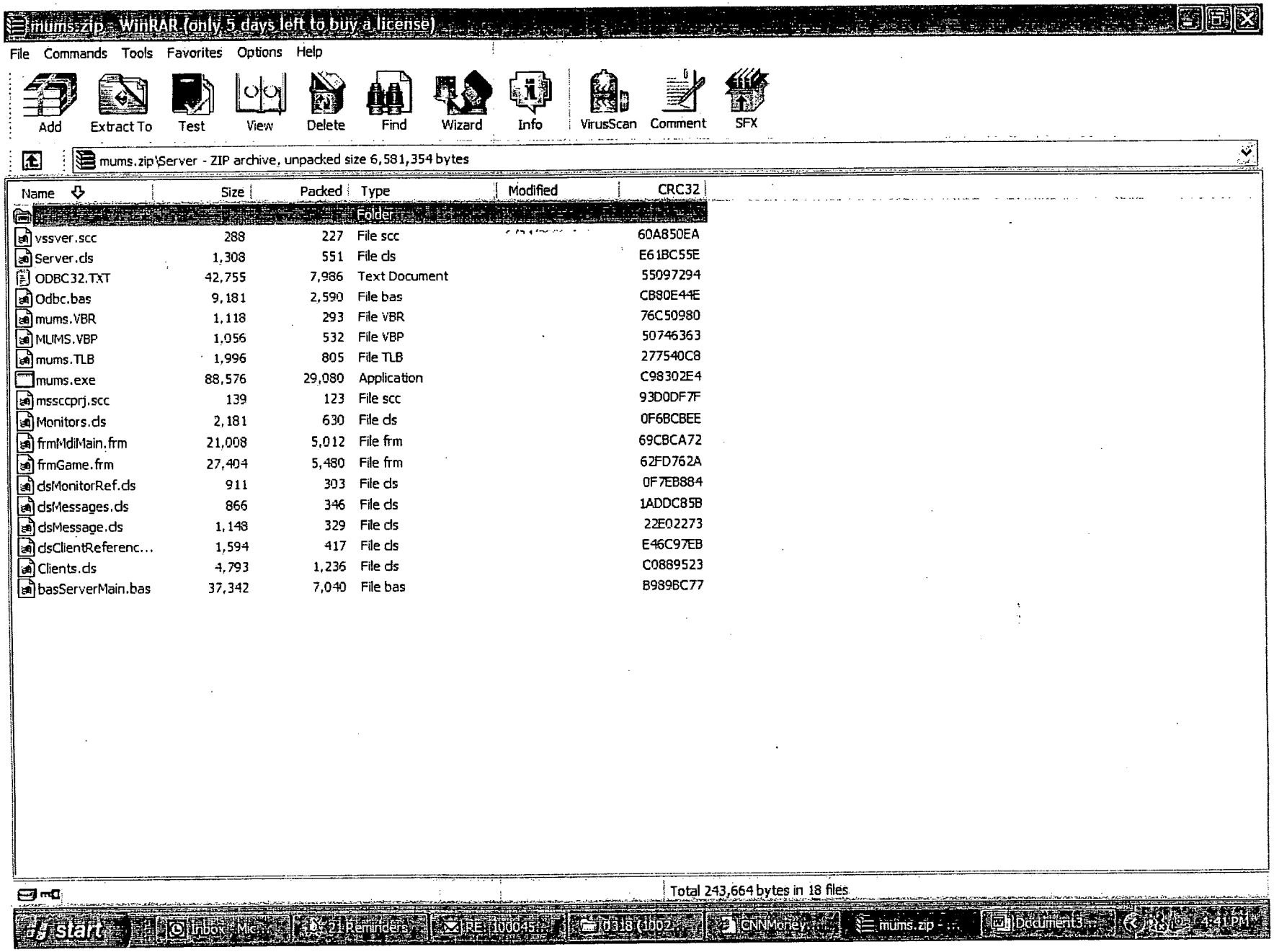
This module will also record a history of all the variables that are being updated in the simulation and output this history as a data set.











MUMS

Script Users Guide

Hewlett-Packard Labs
1501 Page Mill Road, MS 4U-1
Palo Alto, CA 94304-1126

Table of Contents

SCRIPT BASICS

1. INTRODUCTION

In this section, we describe the core elements of a MUMS script and construct an example of a script, in the last chapter, that includes the basic elements. There are a few central elements of scripts, including a set of players, variable declarations, players' variable inputs, and display statements. No key words in Mums are case sensitive, you can use any convention that makes your scripts more readable. This includes all math functions and statements in Mums.

1.1 Comments

To add comments to your script file use the double forward slash "//". Comments can be placed on a line by themselves or at the end of a line containing valid script. Anything after a double forward slash is ignored during the execution of the script.

1.2 Players

The first statement in a script file is the definition of a set of players. The syntax for player declaration is

```
Players PlayerName1, PlayerName2, PlayersName3;
```

where PlayerNamei can be any string that isn't a reserved name. There is one predefined array "Player[]" containing a list of all player names. This is a zero-based list of all players in a game that goes to n -1 players, where n is the number of players declared with the **Players** keyword. The order the players appear in the list is the same order they were defined using the **Players** keyword. A variable can be used for the index in the array and the player array can be used as you would any string array.

1.3 Variables

As with any scripting language, Mums allows you to define variables that can be used in the script. Mums supports three variable types: Integers, Strings, and Real Numbers. In addition, predefined variables allow you to retrieve specific information about what is happening during the execution of the game. All variables, including the predefined variables are case sensitive.

1.4 Functions

Functions allow you to reduce the amount of programming by predefining certain steps. These functions can be used as often as desired anywhere inside a stage. Although you can pass as many variables, including arrays, into a function, you can only return a single value. All variables are passed into a function "By Value", this means you can not change the value of a variable that was passed into a function. All function names are case sensitive.

1.5 Stages

Stages identify a block of script that is executed. In some cases, the execution of the script is passed from stage to stage as if they did not exist. The exception to this, is when you have a real time stage. In this case, execution loops within a stage, from beginning to end and back to the beginning, until the real time execution has completed. Another example where stages are come into play is with the GOTO statement. When a GOTO is encountered execution is sent to the beginning of the stage identified by the GOTO statement.

There is no limit to the number of stages in a script but there must be at least one stage. The END statement is used in a stage to indicate the end of the game or period. In cases where there are multiple periods in a game, when the END statement is encountered the game halts. For a multiple period game, the game automatically restarts at the beginning stage without resetting any of the variables.

1.6 User Inputs

Getting data from users is accomplished by defining INPUT functions. The INPUT statement declares a user-defined function that retrieves data from a user. There are several methods that can be used in getting data from the user, see the INPUT statement for a detailed description on the syntax. How the data is retrieved is determined by the WAIT statement. This can be anything from a single user giving an input to all users having to respond before execution continues. In addition, there are several real time options that allow continue execution while the users are giving their inputs.

1.7 Sending Data to Users

In addition to getting data from a user, it is also necessary to send them information. The DISPLAY statement allows you to send data and setup specific definable fields. Review the DISPLAY statement for a better understanding of the options you have for sending information to users.

WRITING A SCRIPT FILE

5. OVERVIEW

In this section we will discuss how to write a script file that can be executed by MUMS. The script file can be broken down into several defined blocks, in the following sections we will discus the various sections of the script file and give examples of how they would be written.

5.1 Example Script File

This is an example script file that shows some of the features of the MUMS scripting. It is not intended to be a worthwhile game to play, although it does work. It's furnished here so it can be seen how all of the pieces fit together to form a valid script file.

```
// Define Player List
   Players P_A P_B;
// Define Variables
   Integer iAgeA, iAgeB, iWeightA, iWeightB;
   Real rAverageAge, rAverageWeight;
   String sSampA, sSampB;
   String sCharacteristicsA, sCharacteristicsB;
// Define Variables for Color Stage
   Integer bRedA, bBlueA, bYellowA;
   Integer bRedB, bBlueB, bYellowB;
   Integer bColorSet;
   String sColorA, sColorB;
// Define Input Statement
   Input GetData (iTmp1, iTmp2) {
         iTmp1 = txt ( "Enter " & sSampA & " Age" , 20 , 50 , 25 );
         iTmp2 = txt ( "Enter " & sSampA & " Weight" );
   }
// Define Input Statement
   Input GetData2(bTmp1, bTmp2, bTmp3 ) {
         bTmp1 = chk ( "Blue");
         bTmp2 = chk ( "Red");
         bTmp3 = chk ( "Yellow"  );
    }
Stage Start {
         Display ("George", Player) P_A;
         Display ("Martha", Player) P_B;
         Attribute(ON) P_A;
         sCharacteristicsA = "You are a brunette female with 2 children - ages 10 and 14.";
         sCharacteristicsB = "You are a middle aged single male.";
         Display (sCharacteristicsA, Attribute) P_A;
         Display (sCharacteristicsB, Attribute) P_B;
         Display ("AVERAGES and COLORS", Title) P_A, P_B;
         Tutorial("d:\projects\hp mums\sample.txt");
         sSampA = "age";
         sSampB = "weight";
         Display ("Please enter your " & sSampA & " and " & sSampB & " in the boxes below.", Question )
P_A, P_B;
```

```
        GetData (iAgeA, iWeightA) P_A;
        GetData (iAgeB, iWeightB) P_B;
        Wait ( 2);
        rAverageAge = ( iAgeA + iAgeB ) / 2;
        rAverageWeight = ( iWeightA + iWeightB ) / 2;
        Display( "The average age is: " & rAverageAge & "  and the average weight is: " & rAverageWeight
, Question ) P_A, P_B;
}
Stage Colors {
        Display ("Check some combination of colors in the boxes below." , Question ) P_A, P_B;
        Tutorial("d:\projects\hp mums\sample.rtf");
        GetData2 (bBlueA, bRedA, bYellowA) P_A;
        GetData2 (bBlueB, bRedB, bYellowB) P_B;
        Wait (2);
        // Determine color combination for P_A
        bColorSet = 0;
        If ((bBlueA = -1) & (bRedA = -1) & (bYellowA = -1)) {
           sColorA = "Brown";
           bColorSet = -1;
        }
        If ((bColorSet = 0) & (bBlueA = -1) & (bRedA = -1)) {
           sColorA = "Purple";
           bColorSet = -1;
        }
        If ((bColorSet = 0) & (bBlueA = -1) & (bYellowA = -1)) {
                sColorA = "Green";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bRedA = -1) & (bYellowA = -1)) {
                sColorA = "Orange";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bBlueA = -1)) {
                sColorA = "Blue";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bRedA = -1)) {
                sColorA = "Red";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bYellowA = -1)) {
                sColorA = "Yellow";
                bColorSet = -1;
        }
        If (bColorSet = 0) {
                sColorA = "None";
        }
        //  Determine Color combination for player b
        bColorSet = 0;
        If ((bBlueB = -1) & (bRedB = -1) & (bYellowB = -1)) {
                sColorB = "Brown";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bBlueB = -1) & (bRedB = -1)) {
                sColorB = "Purple";
                bColorSet = -1;
        }
```

```
        If ((bColorSet = 0) & (bBlueB = -1) & (bYellowB = -1)) {
                sColorB = "Green";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bRedB = -1) & (bYellowB = -1)) {
                sColorB = "Orange";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bBlueB = -1)) {
                sColorB = "Blue";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bRedB = -1)) {
                sColorB = "Red";
                bColorSet = -1;
        }
        If ((bColorSet = 0) & (bYellowB = -1)) {
                sColorB = "Yellow";
                bColorSet = -1;
        }
        If (bColorSet = 0) {
                sColorB = "None";
        }
        Display("Your color combination is " & sColorA & " while your neighbor chose " & sColorB ,
Question ) P_A;
        Display("Your color combination is " & sColorB & " while your neighbor chose " & sColorA ,
Question ) P_B;
        End;
}
```

MUMS

Installation and Maintenance Guide

Hewlett-Packard Labs
1501 Page Mill Road, MS 4U-1
Palo Alto, CA 94304-1126

Table of Contents

GENERAL

1. INTRODUCTION

This document describes the Installation and Maintenance operations for Mums. You should be familiar with this guide before installing or using Mums to ensure you have setup the system appropriate for your needs. Mums consists of three basic components. The server coordinates the actions of the users and runs the game. The Monitor starts the games and controls the users. Finally, the Client application allows the user to participate in a game. Your installation and operation will be dependent on which option you select during the installation process.

If you are in development mode, it would be appropriate to install all of the components on a single system. Once you have developed and tested your scripts in this environment, you can move them to your main installation where users can participate in a game. For more information on developing scripts see the "Script Users Guide" for more details.

In an operational environment, you should install the server portion on a single machine. You may or may not wish to install the database on the same machine. If you are going to have many users accessing the system at one time, you will get better performance by having the database installed on a separate system. If the number of users is less than 17 you can have both the Mums Server and the database running on the same machine. The client application should always be installed on separate machines that can access the server over a network.

1.1 System Requirements

System requirements will vary depending on how you are installing Mums. In this section, we will discuss the minimum requirements for a Mums implementation. It will be assumed that moderate size scripts are being executed with 15 users attached to the server. The exact configuration for your implementation will depend on how complex the scripts are that you are executing and how many games are in simultaneous play.

1.1.1 Network Installation

The most common implementation is a network installation. Where you have numerous clients connected via a network to the Mums Server. Although there is no physical limit to the number of clients and/or monitors that can be connected to a Mums Server or the number of games under play, these factors do effect performance.

The minimum system requirements defined in this section assume there are five games under play with three users connected to each game. If your implementation exceeds this or you find performance is slower than you expect you will need to get faster hardware. Generally there is a balance between the number of processors used and the amount of

memory required by Mums. You should run a few test games, monitoring the memory and CPU usage to ensure you have adequately sized the system for the games under play.

1.1.1.1 Server Requirements

The MUMS software runs on the Microsoft Windows NT 4.0, or higher, operating system. The following are the minimum requirements for Mums:

1) Processor – Dual 400MHz Processors.
2) Memory – 256MB of RAM.
3) Disk Space – 32MB plus the space setup for the database.
4) Database – SQL Server Database.
5) Network Protocol – TCP/IP or IPX.

MUMS will run on several network protocols, including TCP/IP, IPX, and NetBUEI. As part of the installation of MUMS, a program folder named Hewlett-Packard is created on the Start Menu under Programs. This folder includes the "Remote Automation Connection Manager" (RAC Manager). Use the pull down menu for Network Protocols on the RAC Manager to see a complete list of Network Protocols.

1.1.1.2 Client Requirements

The Mums Client Application is usually installed on a separate system, it should have Windows NT Workstation 4.0, or higher, operating system installed on it. The following are the minimum requirements for the Mums Client:

1) Processor – Single 300MHz Processor.
2) Memory – 48MB of RAM.
3) Disk Space – 16MB
4) Database – None.
5) Network Protocol – Same as Mums Server.

1.1.1.3 Monitor Requirements

The Mums Monitor Application is usually installed on a separate system, it should have Windows NT Workstation 4.0, or higher, operating system installed on it. The following are the minimum requirements for the Mums Monitor:

1) Processor – Single 300MHz Processor.
2) Memory – 32MB of RAM.
3) Disk Space – 16MB
4) Database – None.
5) Network Protocol – Same as Mums Server.

1.1.2 Local Installation

The local installation involves installing all components on a single system. This configuration is usually used for development purposes where you want to run multiple clients and the server on a single system. During the development of scripts you may want to have access to all the components to test run the scripts.

Local installation provides a Microsoft Access Database for storing all data generated by a game. Although Microsoft Access is not required for Mums to run, you may want to have it installed if you want to open the database for analysis.

1.1.2.1 Requirements

The following are the minimum requirements to run Mums on a single system. If you are generating a large amount of logged data, you may want to provide additional space for the Microsoft Access Database.

1) Processor – Single 300MHz Processor.
2) Memory – 96MB of RAM.
3) Disk Space – 48MB
4) Database – Microsoft Access.
5) Network Protocol – None Required.

1.2 Installation Instructions

In this section, we will discuss four installation options available for Mums. Which options you choose will depend on the purpose of the installation.

1.2.1 Network Installation

Before running the installation program, ensure SQLServer is setup and is running. See the Section on SQL Server Installation for instructions on configuring SQL Server for Mums.

1.2.1.1 Server Installation

Before installing any other component of Mums, the server must be installed. Other components will need to know the location of the server. To install the Mums Server follow these steps:

1) Browse the file system and locate the disk containing the installation program, open the disk and double click on "setup.exe"
2) During the installation, a dialog will be displayed allowing you to select the components to install. Select the desired components but do not select Local Installation.
3) You will be prompted for the Server Name, enter the machine name of the SQL Server database.
4) Follow the instructions displayed on the screen.
5) This completes the Mums Server installation.

The Mums Server must now be configured so it can be seen by the other components across the network. These steps are only required the first time you install the Mums Server on a system, they are not required for any follow on installations. To configure the Mums Server follow these steps.

1) Using the File Explorer go to the directory where you installed Mums.
2) Open this directory and locate the folder "Programs", right click on it and select "Sharing".
3) On the dialog that comes up, click on the "Share As" option.

4) For the "Share Name", enter "Mums Server".
5) Click on "OK".
6) On the left bottom corner of the screen select "Start", when the window opens up select "Programs" then "Hewlett Packard". Click on the "RAC Manager" application.
7) In the left column of the RAC Manager window select "MiniMums.theHelper", you may have to scroll down to see it.
8) On the right side of the screen you will see "Network Protocol", enter TCP/IP.
9) Click on "Apply"
10) This ends the Server configuration, exit from the RAC Manager. Installation is complete.

1.2.1.2 *Client Installation*

The client application allows the user to interact with the gam in play. To install the Mums Client follow these steps:

1) Browse the file system and locate the disk containing the installation program, open the disk and double click on "setup.exe"
2) During the installation, a dialog will be displayed allowing you to select the components to install. Select the Client component and make sure all other components are not selected.
3) You will be prompted for the Server Name, enter the machine name of the SQL Server database.
4) Follow the instructions displayed on the screen.
5) This completes the Mums Client installation.

The Mums Client must now be configured so it can be seen by the other components across the network. These steps are only required the first time you install the Mums Client on a system, they are not required for any follow on installations. If you are installing the Client on a system that already contains a server, you do not need to complete these steps. To configure the Mums Client follow these steps.

1) Using the "Network Neighborhood" browse the network and find the server where the Mums Server was installed and open it.
2) Double click on "Mums Server".
3) Locate the application "MiniMums.exe" and double click on it.
4) Close the Network Neighborhood window.
5) On the left bottom corner of the screen select "Start", when the window opens up select "Programs" then "Hewlett Packard". Click on the "RAC Manager" application.
6) In the left column of the RAC Manager window select "MiniMums.theHelper", you may have to scroll down to see it.
7) On the right side of the screen you will see "Network Protocol", enter TCP/IP.
8) Click on "Apply"
9) On the RAC Manager Menu bar you will see "Permissions", click on it and select "Allow Remote Activation".
10) Click on "Apply".
11) This ends the Client configuration, exit from the RAC Manager. Installation is complete.

1.2.1.3 Monitor Installation

The monitor application is used to start games and control the game under play. To install the Mums Monitor follow these steps:

1) Browse the file system and locate the disk containing the installation program, open the disk and double click on "setup.exe"
2) During the installation, a dialog will be displayed allowing you to select the components to install. Select the Monitor component and make sure all other components are not selected.
3) You will be prompted for the Server Name, enter the machine name of the SQL Server database.
4) Follow the instructions displayed on the screen.
5) This completes the Mums Monitor installation.

The Mums Monitor must now be configured so it can be seen by the other components across the network. These steps are only required the first time you install the Mums Monitor on a system, they are not required for any follow on installations. If you are installing the Monitor on a system that already contains a Mums Server or a Mums Client, you do not need to complete these steps. To configure the Monitor follow these steps.

1) Using the "Network Neighborhood" browse the network and find the server where the Mums Server was installed and open it.
2) Double click on "Mums Server".
3) Locate the application "MiniMums.exe" and double click on it.
4) Close the Network Neighborhood window.
5) On the left bottom corner of the screen select "Start", when the window opens up select "Programs" then "Hewlett Packard". Click on the "RAC Manager" application.
6) In the left column of the RAC Manager window select "MiniMums.theHelper", you may have to scroll down to see it.
7) On the right side of the screen you will see "Network Protocol", enter TCP/IP.
8) Click on "Apply"
9) On the RAC Manager Menu bar you will see "Permissions", click on it and select "Allow Remote Activation".
10) Click on "Apply".
11) This ends the Monitor configuration, exit from the RAC Manager. Installation is complete.

1.2.2 Local Installation

The local installation allows you to install all of the Mums components on a single system. In addition, a Microsoft Access Database is used to log all data generated by Mums. This allows stand-alone operation of Mums without any network connectivity. This type of installation is ideal for individuals who will be developing scripts to be run under Mums.

To install Mums as a local installation follow these steps:

1) Browse the file system and locate the disk containing the installation program, open the disk and double click on "setup.exe"
2) During the installation, a dialog will be displayed allowing you to select the components to install. Select all the components including Local Installation.

3) Follow the instructions displayed on the screen.
4) This completes the Mums local installation.

SQL SERVER INSTALLATION

2. INTRODUCTION

When operating Mums across a network it uses a SQLServer database for tracking statistics for games under play. An analysis can be performed on this data once the games have completed. The SQLServer database should be installed and configured after Mums is installed. The installation of Mums will provide a file necessary for building the Mums database.

2.1 Requirements

Before installing SQLServer, Microsoft Windows NT Server version 4.0 or later must be installed. Also, ensure network connectivity exists before starting the SQLServer installation.

2.2 SQLServer Installation

Installation of SQLServer is straight forward, run the "setup.exe" program on the SQLServer CD. Follow the installation instructions until you get to the network option. Click on the Network button and select TCP/IP as an additional network option. Ensure you have identified sufficient licenses for use by Mums. Mums will use one license plus one license for each user that will be connected to the Mums Server.

2.2.1 Configuring SQLServer for Mums

The following steps are required to setup the Mums database and it assumes that SQLServer is installed and running. The following steps should be followed in sequential order.

2.2.2 Starting SQL Enterprise Manager

To start the SQL Enterprise Manager go to the "start" button in the lower left corner of the screen. When the window opens up, select "Program" then "SQL Server" and finally click on "SQL Enterprise Manager".

2.2.3 Register the Server

Once the "SQL Enterprise Manager" has started, perform the following operation if you do not see the name of your system appear on the left side of the screen.

1) Click on the "File" menu of the "SQL Enterprise Manager"
2) Select "Register Server" from the list.
3) In the "Server Name" box, enter your server name, such as "Willow".
4) For "Login ID" enter the system administrators login name. The default system administrator login name for SQLServer is "sa", if the defaults were taken during the installation then this will be the login ID that should be used.

5) Enter the password for the system administrator login. If the defaults were taken during the installation this field should be left blank.
6) Click on the "Register" button to register the Server name with SQLServer.

2.2.4 Creating a Mums Database

The following steps are required to create a database for Mums. All of these actions are performed in the "SQL Enterprise Manager".

2.2.4.1 Create a Database Devise

Before a database can be created, SQLServer requires that a database device exist. This device is where the database will reside and the database can not be larger than the device used to hold it. If it is necessary to expand the size of the database at a later time, this can be done by making the database device larger and then expanding the database size.

Because of the flexibility that exists in logging information in Mums determining the database size can be difficult. It is possible to log up to 1KB worth of data per logging transaction when a script is run. In addition, an additional 1KB of data can be logged each time a specific user action is invoked. In some cases a game can be played for extended period of time which involves numerous records in the database. The best way to determine the optimum database size is through experience you gain in running experiments and how often you clean up the data in the database. The following database size is based on an aggressive use of logging using lengthy games. After running several games you should review the database usage and make changes to it as appropriate.

To create a database device do the following:

1) Expand the name of your computer. For this explanation we will assume the name of the Computer is "Willow", whenever "Willow" is used replace it with the name of your Server. In this case you will see the name "Willow" appear on the right side of the "SQL Enterprise Manager", click on the plus (+) sign to expand it. You will now see several items listed under Willow.
2) Right click on the "Database Devices" item.
3) Select the "New Device" Item.
4) In the screen that comes up:
   a) Enter "Mums" for the device name.
   b) Enter 250MB for the size of the device.
   c) Click on "Create New", this screen will now be removed.
5) It is now time to create a device to hold the database log file. Right click on the "Database Devices" item again.
   a) Enter "MumsLog" for the device name.
   b) Enter 20MB for the size of the device.
   c) Click on "Create New", this screen will now be removed.

2.2.4.2 Create a Database

Once the database device has been created, the database is created using the new device. The following steps are used to create a database.

1) Right click on "Databases" under "Willow".
2) Select "New Database" from the list.

3) A screen will come up for creating new databases, enter the following information.
   a) For the database name enter "Mums".
   b) For the database device enter "Mums".
   c) For the Log Device name enter "MumsLog".
   d) Click on "Create New", this will create the new Mums database and the window will disappear.

2.2.5 Creating a Mums Login Account

The Mums Server and the Clients access the Mums database using a single account. This account must be created for these applications to access the database. The following steps are required to create the account:

1) Right click on the "Logins" item under "Willow".
2) In the window that opens up do the following:
   a) Enter "Mums" for the login name.
   b) Enter "mums" for the password.
   c) The default language is "default"
   d) In the Database Access box, select "Mums".
   e) Set Permit to "default".
   f) Set User to "Mums"
   g) Set Group to "Public"
   h) Set Alias to "None"
   i) Click on "Add", this will add the new user and the window will close.

2.2.6 Create the Mums Database Tables

Now that the database has been created it is necessary to create the tables in the database that Mums will use to log data to. These tables can be used at a later time to retrieve data from to perform an analysis of the games that have been played. A common way to access a SQLServer database for analysis is to link it to a Microsoft Access Database. For instructions on how to connect Access to SQLServer see the section on Linking Access. In addition to connecting an Access database, the SQLServer database can be connected to using an application. See the appropriate programmers manuals on how to accomplish this.

The following steps create the Mums database tables:

1) Under "Willow" open the "Databases" item on the left side of the "SQL Enterprise Manager".
2) Left click on the "Mums" database.
3) On the toolbar, select the "SQL Query Tool" icon, this will open the SQL Query window.
4) In the window, click on the "Query Tab".
5) On the tool bar for the "SQL Query Tool" window, click on the "Open File" icon.
6) In the Open File dialog that has opened, browse the file system to locate the "tables.sql" file. This file is located in the "Programs" directory under the directory Mums was installed.
7) Once the "tables.sql" file has been opened, you will see several lines of SQL script loaded into the query window. This script will create all of the tables for Mums.

8) On the "SQL Query Tool" toolbar click on the "Execute Query" icon. This will cause the SQL statements loaded into the window to be executed.
9) Click on the "Results" tab in the "SQL Query Tool" window. You should see a message stating "Non-clustered ... being rebuilt", this verifies that Mums database tables have been successfully created.
10) Close the "SQL Query Tool" window.

2.2.6.1 *Database Verification*

Once the database tables have been created it is good to verify it. The following steps show you how to verify the tables have been correctly created:

1) Under "Willow" open the "Databases" item.
2) Under "Databases" open the "Objects" item.
3) Under "Objects" open the "Tables" item.
4) You should see the following tables listed:
   a) Commands
   b) ErrorLog
   c) MsgLog
   d) WriteLog

2.2.6.2 *Set SQL Permissions*

The permissions determine how SQLServer will allow users to access the database. To set the permissions follow these steps:

1) Under "Willow" open the "Databases" item.
2) Under "Databases" open the "Objects" item.
3) Under "Objects" open the "Permissions" item.
4) Select "Object"
5) You can now close all open windows.

2.2.7 Setup Mums Data Source

For the Mums server to access the SQLServer Database, it is necessary to setup a data source on the Server machine. The more advanced users can setup the data source on the server machine, following these instructions, and have the database reside on another machine. Inexperienced users should follow these steps.

1) Click on the "Start" button in the lower left corner of the screen.
2) When the window opens up, select "Settings".
3) When the new window opens up, select "Control Panel".
4) In the control panel, double click on "ODBC".
5) Click on the "System DSN" tab.
6) Click on the "Add" button.
7) Scroll down if necessary and find the name "SQL Server" and double click on it. This will cause another window to open up.
8) For the Data Source Name enter "Mums".
9) In the Description box enter "Mums Server DB Access".
10) For the Server select "(local)".
11) Click on the Options button, when the window expands enter "Mums" for the Database Name.
12) Click on "OK", this will cause the window to close.
13) Close all windows, this completes the Mums SQLServer database installation.

2.3 Database Maintenance

During the operation of Mums, the contents of the database will grow over time. To avoid having the database fill up and rendering Mums inoperable, it will be necessary to perform periodic maintenance on the database. In this section, we will discuss how you can view the amount of data in the database and how to setup automatic database maintenance. As with all critical data, you should have a regular backup procedure in place to restore data in the case of a catastrophic event happening to the hardware. See your system administrator to determine how you should best backup your data.

2.3.1 Viewing Database Usage

As Mums is used it is good practice to determine how fast you are consuming database space until you have established a pattern of usage. This will help reduce the risk of running out of database space during a critical run of Mums.

You will need to start the SQL Enterprise Manager to view the database usage. Go to the "start" button in the lower left corner of the screen. When the window opens up, select "Program" then "SQL Server" and finally click on "SQL Enterprise Manager". Follow the steps below to determine how much space is left in the database:

1) On the left side of the "SQL Enterprise Manager", select the name of the server where the Mums database is installed.
2) Expand the "Databases" item in the list.
3) Right click on the "Mums" database and choose "Edit".
4) A window will open showing several attributes of the database. Two items are of concern, this first is the "Data Space Available" and the other is "Log Space Available". These two items show how much space has been consumed within the database. Ensure there is sufficient space available to continue running Mums.
5) Close all of the windows. If there is insufficient space, either make the database larger or remove some of the records in the database.

2.3.2 Automation of Database Maintenance

LINKING MICROSOFT ACCESS

3. OVERVIEW

Often it is necessary to perform analysis on the data generated by Mums. One method of doing so is accessing the data via a Microsoft Access Database. Microsoft Access gives you a straightforward method of creating reports and developing simple analysis tools. In this section, we will discuss how to connect an Access Database across the network to the Mums Database.

3.1 Creating a Mums Data Source

For Microsoft Access to access the Mums SQLServer Database, it is necessary to setup a data source pointing to the SQL Server database. The following steps explain this procedure:

1) Click on the "Start" button in the lower left corner of the screen.
2) When the window opens up, select "Settings".
3) When the new window opens up, select "Control Panel".
4) In the control panel, double click on "ODBC".
5) Click on the "System DSN" tab.
6) Click on the "Add" button.
7) Scroll down if necessary and find the name "SQL Server" and double click on it. This will cause another window to open up.
8) For the Data Source Name enter "Mums Server DB".
9) In the Description box enter "Mums Server DB".
10) For the Server enter the name of the Server the Mums SQLServer Database is installed on.
11) Click on the Options button, when the window expands enter "Mums" for the Database Name.
12) Click on "OK", this will cause the window to close.
13) Close all windows, this completes the Mums SQLServer database installation.

3.2 Creating an Access Database

The Microsoft Access Database you create can continually be used to retrieve data from and perform analysis on the Mums Database. When creating the database, you should give it a name and place it in a location you can remember. To create the database follow these steps:

1) Open Access, on the window that comes up select "Blank Database".
2) A "File New Database" window will pop up, using the top window browse the file system to locate where you would like to have the new database saved. In the "File Name" box, enter the name of the new database and click the "Create" button.
3) The database will be created and you will have a window open with several tabs, the "Tables" tab will be blank.
4) Click on the "New" button.

5) A new window will pop up with several options, select "Link Table" and click OK.
6) In the new window, go to the "Files of Type" drop down list box and select "ODBC( ) Databases.
7) Select "Data Source" and then select "Machine Data Source".
8) Select the data source you created in section 3.1, "Mums Server DB"
9) When prompted for the user name and password, enter "Mums" for the user name and "mums" for the password.
10) In the window that comes up, select "dbo.ErrorLog" and "dbo.WriteLog then click on the "OK" button.
    a) The dbo.ErrorLog table shows all errors that have occurred during the operation of Mums. This table is not required but is good to review occasionally if you are interested in what errors occurred during the operation of Mums.
    b) The dbo.WriteLog table contains all of the data that is logged during the operation of Mums. This is the table you will perform analysis on.

MUMS DB SCRIPT

4. OVERVIEW

The following script (tables.sql) is used to create the tables in the Mums database. This script is for reference only and can be used by more advanced users who want to access the data directly.

4.1 SQL Script

```
/* Microsoft SQL Server - Scripting                    */
/* Server: YOURSERVER                                  */
/* Database: Mums                                      */
/* Creation Date 1/22/97 9:22:41 AM                    */

/****** Object:  Table dbo.Commands    Script Date: 1/22/97 9:22:47 AM ******/
if exists (select * from sysobjects where id = object_id('dbo.Commands') and sysstat & 0xf = 3)
        drop table dbo.Commands
GO

/****** Object:  Table dbo.ErrorLog    Script Date: 1/22/97 9:22:47 AM ******/
if exists (select * from sysobjects where id = object_id('dbo.ErrorLog') and sysstat & 0xf = 3)
        drop table dbo.ErrorLog
GO

/****** Object:  Table dbo.MessageLog    Script Date: 1/22/97 9:22:47 AM ******/
if exists (select * from sysobjects where id = object_id('dbo.MessageLog') and sysstat & 0xf = 3)
        drop table dbo.MessageLog
GO

/****** Object:  Table dbo.Commands    Script Date: 1/22/97 9:22:48 AM ******/
CREATE TABLE dbo.Commands (
        Command char (5) NOT NULL ,
        Description varchar (255) NULL ,
        Parameters varchar (255) NULL ,
        Actions varchar (255) NULL ,
        Notes varchar (255) NULL
)
GO

 CREATE  UNIQUE  CLUSTERED  INDEX PrimaryKey ON dbo.Commands(Command)
GO

/****** Object:  Table dbo.ErrorLog    Script Date: 1/22/97 9:22:51 AM ******/
CREATE TABLE dbo.ErrorLog (
        ID numeric(18, 0) IDENTITY (1, 1) NOT NULL ,
        ErrNumber int NOT NULL ,
        ErrorMessage varchar (56) NOT NULL ,
        ClientID varchar (26) NOT NULL ,
```

```
        FunctionName varchar (26) NOT NULL ,
        TimeOccurred datetime NOT NULL ,
        ProgramSource varchar (8) NOT NULL
)
GO

 CREATE  INDEX ClientID ON dbo.ErrorLog(ClientID)
GO

 CREATE  UNIQUE  CLUSTERED  INDEX PrimaryKey ON dbo.ErrorLog(ID)
GO

/****** Object:  Table dbo.MessageLog    Script Date: 1/22/97 9:22:51 AM ******/
CREATE TABLE dbo.MessageLog (
        ID numeric(18, 0) IDENTITY (1, 1) NOT NULL ,
        ClientID varchar (26) NOT NULL ,
        MessageCode char (5) NOT NULL ,
        MessageTime datetime NOT NULL ,
        Parameters varchar (255) NULL ,
        GameID varchar (26) NULL
)
GO

 CREATE  INDEX ClientID ON dbo.MessageLog(ClientID)
GO

 CREATE  INDEX Message ON dbo.MessageLog(MessageCode)
GO

 CREATE  UNIQUE  CLUSTERED  INDEX PrimaryKey ON dbo.MessageLog(ID)
GO

/****** Object:  Table dbo.WriteLog    Script Date: 4/2/97 trl ******/
if exists (select * from sysobjects where id = object_id('dbo.WriteLog') and sysstat & 0xf = 3)
        drop table dbo.WriteLog
GO

/****** Object:  Table dbo.WriteLog    Script Date: 4/2/97 trl ******/
CREATE TABLE dbo.WriteLog (
        IID numeric (18,0) IDENTITY (1, 1) NOT NULL,
        GameID varchar (64) NOT NULL,
        PlayerID varchar (64) NULL ,
        Period int NOT NULL ,
        MessageTime datetime NOT NULL ,
        TimeInGame int NOT NULL,
        LogType varchar(64) NULL,
        LogField1 varchar (64) NULL,
        LogField2 varchar (64) NULL,
        LogField3 varchar (64) NULL,
        LogField4 varchar (64) NULL,
        LogField5 varchar (64) NULL,
        LogField6 varchar (64) NULL,
        LogField7 varchar (64) NULL,
        LogField8 varchar (64) NULL,
        LogField9 varchar (64) NULL,
```

```
        LogField10 varchar (64) NULL,
        LogField11 varchar (64) NULL,
        LogField12 varchar (64) NULL,
        LogField13 varchar (64) NULL,
        LogField14 varchar (64) NULL,
        LogField15 varchar (64) NULL
)
GO

 CREATE  INDEX GameID ON dbo.WriteLog(GameID)
GO

 CREATE  INDEX PlayerID ON dbo.WriteLog(PlayerID)
GO

 CREATE INDEX GamePlayerPeriod ON dbo.WriteLog(GameID, PlayerID, Period, MessageTime)
GO

 CREATE  UNIQUE  CLUSTERED  INDEX PrimaryKey ON dbo.WriteLog(IID)
GO
```

Minimum Advertised-Price Policy Rules and Retailer Behavior: An Experiment by Hewlett-Packard

Gary Charness • Kay-Yut Chen
Department of Economics, University of California, Santa Barbara, Santa Barbara, California 93106
Hewlett-Packard Laboratories, P.O. Box 10301, Palo Alto, California 94303-0890
charness@econ.ucsb.edu • kychen@exch.hpl.hp.com

This paper was refereed.

We tested the effects of various policy rules on retailer behavior in laboratory experiments conducted at Hewlett-Packard Laboratories. Our experimental design models the multifaceted contemporary market for consumer computer products and is quite complex, but we found that participants can make effective decisions and that their behavior is sensitive to variations in policies. Based on our results, Hewlett-Packard changed its policies; for example, it made the consequences for violations forward looking as well as backward looking. This line of research appears promising for complex industrial environments.
(*Industries: computer-electronic. Economics: experimental.*)

Manufacturers operating in the contemporary market for technology products face a daunting task in designing effective incentives for their retailers. Channels of distribution are diverse, with new channels emerging, and demand fluctuations, market exposure, advertising, stocking, and product life-cycles are uncertain. The behavior of retailers is a critical element in whether a manufacturer achieves its business goals.

Ideally, a firm would like to have a test market to determine the effects of changing its policies toward its retailers, since blindly adopting new policies in billion-dollar markets may be less than optimal. However, test markets may not be feasible in a technology market without geographic moorings; at best, test markets are expensive. While computer simulations are useful, their applicability depends on assumptions about the decisions human agents make in the field. An alternative testing method is to model retailers' choices in laboratory experiments. Economics experiments are often used in academic research to test polices; they can also be applied to business.

Such experiments have been used to examine behavior in laboratory market contexts since the publication of the seminal works of Edward Chamberlin (1948) and Vernon Smith (1962, 1964). Strong indications of external validity (applicability to the field environment) exist for behavior observed in laboratory markets, even when the environment is highly complex. For example, Vernon Smith and Charles Plott (Plott 2002, McCabe et al. 1990) pioneered smart markets to examine complex interdependent environments. In a smart market, a computerized dispatch center applies optimizing algorithms to the diverse and decentralized bids of buyers and the offers of producers and transporters to yield prices and allocations. Smith and Plott found that experimental markets can produce repeatable and predictable results. Researchers have used the methodology of experimental economics to test alternative policies in such areas as emissions trading, natural-gas pipelines, electric-power-transmission networks, transportation, and water distribution in California (Brewer and Plott 2002,

0092-2102/02/3205/0062$05.00
1526-551X electronic ISSN

Cason 1995, Cason and Plott 1996, McCabe et al. 1990, 1991, Plott 1997, 1999, Rassenti et al. 1994).

Hewlett-Packard Company (HP) has recognized the potential of this methodology as a decision support tool; Hewlett-Packard Laboratories (HP Labs), the research arm of HP, began an experimental economics program in 1994. The firm recognizes the importance of both economic modeling and experimental methods as tools to support business decisions. Its strategy is to develop experimental models that closely mirror specific HP businesses and to then employ these models to isolate and evaluate the effects of specific policies.

Throughout its six-year history, the HP experimental economics program has performed research and developed applications in several areas, including channel management, forecasting, and electronic markets. Several HP divisions have recognized that experimental studies of the behavior of sales channels under different sets of contractual terms and business policies can provide extremely useful information before they implement such terms and policies in the field.

We developed an experimental application in the channel-management area. We are doing additional research, using game-theoretic analysis, on these issues in collaboration with John Ledyard at the California Institute of Technology. HP conducts much of its consumer business through retail channels. The distribution channels for its products include national retailers, regional retailers, mass merchant firms, clubs, and Internet retailers. Each type of retailer may have its own success metrics or business goals, which may or may not be consistent with those of HP. For example, at the time of our experimental sessions many observers felt that Internet retailers were not concerned with current profitability, as these retailers often sold to consumers at or below cost in an attempt to increase their market share. HP is concerned with the financial viability and market share of its retailers, if only because they affect its own market share and profitability.

HP Labs began an experimental economics program in 1994.

HP uses policies to govern its relationship with its retailers, for example, return policies, price-protection policies that provide credit to the retailers corresponding to manufacturers' price fluctuations, and benefits or penalties contingent on retailer compliance with a minimum advertised-price (MAP) policy. To design effective policies consonant with its business objectives, HP must understand the implications of these policies on retailer behavior.

In this series of experiments, we studied the behavior of retailers with respect to the common industry practice of setting a minimum advertised price, a lower bound on the price a retailer can advertise for a particular product. If a retailer complies with this directive, the manufacturer typically provides it with market-development funds, which it can use to advertise the manufacturer's products. If it does not comply, it usually faces penalties. Because thousands of products are involved, MAPs are usually not enforced by legal contracts. Punishment can range from refusing to ship a product to the retailer to eliminating or reducing the amount of market development funds provided.

HP sets MAPs because it might lose market share if retailers perceived that price competition for HP products was too intense. Yet it is not clear which form of MAP (if any) is best and which enforcement policies are effective. Thus, HP wants to know what effect eliminating or modifying MAP policies would have on its market share and its retailers' profitability. An effective policy should also take into account such factors as the short life cycle of products in this market. Because it is not feasible to isolate a test market of retailers, the laboratory is an attractive alternative for investigating the impact of various policies. We conducted laboratory experiments to investigate the effects of various MAP policies on retailers' behavior and profitability, and on HP's market share.

Business Constraints and Design Limitations

We used the standard methodology of experimental economics. We brought participants into a lab and assigned them roles as retailers. They interacted with other retailers by setting prices, choosing advertising expenditures, and receiving rewards and penalties as specified by the extant policies. We gave them accurate information about the game, and told them how their

actual monetary rewards depended on their aggregate performance over the course of the session. We preserved experimental anonymity with respect to roles and payment, and we used no deception. Nevertheless, business-decision research differs from academic research. First, the experimental design went through a validation process, in which HP industry experts played the experimental game and offered feedback. Second, the business environment imposed constraints in terms of experimental design, procedures, and timetable.

HP Labs developed in-house experimental economics capabilities instead of relying on academic institutions for consultants because business considerations make such consultation impractical. Business decisions must be made in a timely fashion, even if they are made with less than perfect information. HP typically develops its potential business in three to six months, depending on the cycle of contract and policy decisions. Thus, we often design our experiments in the expectation that redesign and repetitions are unlikely, except in the most critical situations. Academic researchers generally want to establish statistical significance, necessitating replications and increasing the turnaround time.

Also, in industrial settings, it may be that no tractable theory on the research questions of interest exists, and time may prohibit developing a theoretical model that could point to specific issues to test. Because time limitations meant we could not explore the parametric space fully and because HP wished to preserve the complexity of the field environment, we tried to include as many of its features (that is, stochastic supply, demand, and delivery times, residual advertising effectiveness, and price reputation) as possible in the experiment. Our experimental environment was therefore quite complex.

We cannot vouch for the robustness of the results.

This design philosophy runs counter to standard academic experimental practice, where researchers prefer the simplest design that can encompass the modeling issues at hand. As a result, we cannot vouch for the robustness of the results. For example, if we observe some participants exploiting a policy in a certain way, we have no idea whether this behavior is an equilibrium strategy, a likely occurrence, or something that will be eliminated in the long run. However, from a business point of view, identifying such exploitation is unquestionably useful, whether or not it is the optimal strategy for a retailer. In effect, we are employing subjects to find flaws in proposed policies.

An obvious disadvantage of combining a complex design with a lack of repetition is our resulting inability to identify cause and effect. We did not control most of the many variables because of time pressure and because management does not consider it a high priority. Academic researchers may not see this approach as satisfactory; we cannot clearly attribute the findings to specific variables, as many of these were being changed simultaneously. Strictly speaking, from the standpoint of statistical analysis, we have only one observation for each session.

Nonetheless, we felt this research strategy was the most effective for obtaining the information requested in the time allocated. HP was interested in the result of changing a policy but was rather indifferent about what caused the result. The data indicate that our results are consistent with real-world observations.

Experimental Design

In our laboratory market, we attempted to model the natural setting for HP retailers. Each participant represented a retailer, while demand was computer-simulated using a model. We had heterogeneous firms interacting repeatedly in competing for consumer demand for products differentiated by price and manufacturer. Retailers made decisions about stocking, advertising, and pricing. Each (simulated) consumer considered the best price available when deciding whether to buy a product but was only aware of the products and prices to which it was exposed. A retailer's demand could also be sensitive to its reputation for pricing, relative to other retailers.

Seven differentiated retailers interacted in each of our sessions. They were intended to represent national firms, PC Direct/Mail Order companies, mass merchants, clubs, and Internet retailers. PC Direct companies are ones that sell HP printers with their PCs.

Each retailer chose a price for each product in each period and competed for some percentage of the potential market for the products. Most firms could increase this percentage by advertising, although each type of retailer had a maximum exposure percentage and advertising yielded diminishing marginal returns. Most retailers also had to make inventory decisions, with the cost of holding excess inventory balanced against a negative reputation if a retailer failed to meet most of the demand for a product. The timing of deliveries to the retailers was stochastic.

We computer-simulated consumer demand using a random utility multilevel logit model (Dubin 1998, McFadden 1976) adapted to the HP environment by Steven Gjerstad and Jason Shachat. This model treats each product as a collection of attributes (such as price, brand, retailer, speed, and memory). When assessing a potential product choice, each consumer assigns a different weight to the value of each attribute, and the model adds these values together to determine that consumer's score for the product. The probability that the consumer purchases a product increases with this score, and the probability that any one product is selected is the estimated market share of that product.

HP was interested in the result of changing a policy but indifferent about what caused the result.

The stochastic market size lies within a range known to the retailers, who also receive a signal that further limits this range at the beginning of a period.

Retailers can sell products offered by HP and by competing manufacturers. These products vary by retailer cost and by manufacturer policies on product returns and advertising. We evaluate different retailers using diverse measures that reflect the contemporary business goals of the different categories of retailer. These measures include various combinations of gross profit, net income, revenue, and GMROII (GMROII is based on the product of revenue and the ratio of gross profit to total inventory value for the past four periods; this is a common performance measure in this industry). The model incorporates product obsolescence through a life-cycle assumption—some products get phased out and others take their place, with retailers receiving notice five periods in advance.

Inventory control is a crucial aspect of the natural retailer environment. Most retailers (although not all) need to stock products to be able to sell them. However, it is usually costly to carry excess inventory. In addition, while a retailer may place an order for products, the actual shipment date is uncertain. Further, supplies may be short at any particular time. Retailers must consider all of these factors when making stocking decisions; a retailer who cannot meet existing demand develops a negative reputation for service, which negatively affects subsequent demand.

Finally, advertising clearly affects demand and must be considered, particularly because advertising policy is the control variable in the experiment. A retailer has some minimum level of market exposure even without any advertising. However, advertising increases market exposure in a nonlinear fashion, until it saturates the market for the retailer. While a firm may be free to advertise any price it likes, violating manufacturer mandates concerning minimum advertised price jeopardizes the advertising funds potentially available from the manufacturer. Manufacturers employ several schemes to punish violations.

The natural market is very complex and even chaotic, with new types of retailers growing in importance. Planners within manufacturing firms must somehow formulate policies that take important marketplace features into account without making decisions so difficult that the results are arbitrary.

Experimental Procedure

We conducted our first set of sessions in September 1999. We used the insights obtained in September to modify our design for our second set of sessions in February 2000. (Detailed experimental instructions are available upon request; we omit the fine detail of our calibrations and models to protect intellectual property.)

We recruited participants by sending an e-mail message to Stanford University interest groups. Most of our subjects turned out to be graduate students. Because of the complexity of the environment and the need for participants to make several decisions each

Retailer #	Must stock?	Can advertise?	Minimum % market exposure	Maximum % market exposure	Evaluation method
1	yes	yes	30	100	70% GMROII, 30% Net income
2	yes	yes	30	100	Gross profit
3	yes	yes	30	70	70% GMROII, 30% Net income
4	yes	yes	30	70	70% GMROII, 30% Net income
5	yes	yes	30	50	100% GMROII
6	yes	no	40	40	70% GMROII, 30% Net income
7	no	yes	10	30	Revenue

Table 1: In our September experiments, participants played the roles of seven very different types of retailers. They differed in many aspects from their reach in the market (min/max percent market exposure) to whether they stocked and held their own inventories (some retailers fulfilled orders through a third party and held no inventory).

period, our initial sessions were quite lengthy (we have now developed a design that facilitates much shorter sessions).

In establishing pay rates for participants, we tried to calibrate expected earnings to about $18 per hour (including a show-up fee), and actual earnings ranged from $10 to $25 per hour. However, we could not make any guarantees about pay, and the time requirement made it rather difficult to fill the sessions. Participants were paid a show-up fee of $25, and their remaining earnings were based on their profitability. We used a dollar conversion rate that varied by the type of retailer.

The participant-retailers viewed information on a series of six screens:

(1) The order screen offered them an opportunity to make purchases and listed past-period pricing and margins for each retailer, how much was spent on advertising for each product in the upcoming period, and inventory and ordering information.

(2) The advertising screen again presented pricing and inventory information and also stated the amount available for advertising. Participants chose advertising expenditure for each product. Advertisements ran four periods after the space was reserved, although the retailer chose the advertised price in the period that the advertisement appeared. There was a two-period lag between the choice of advertising expenditures and the appearance of the advertisement (except for retailer 7, who had no lag). Retailers could advertise only with the advertising funds provided in an initial endowment and later supplemented by manufacturer funds based on product purchases.

(3) Retailers chose selling prices (these were also the advertised prices for that period) on the pricing screen, which again listed pricing and inventory information and also indicated any pricing restrictions. There were no restrictions on the price per se; if no advertisement appeared for a product in a period, no MAP violation would occur, regardless of the selling price.

(4) The price-control-and-ad screen showed the advertising funds earned from the shipments received in that period, the amount lost in that period because of a MAP violation, and the number of periods remaining in the MAP penalty.

(5) The supply, demand, and return screen showed the retailer's demand for each product for the period. If units had been ordered but supply was rationed, this was indicated. If a stock-out penalty (for servicing less than 50 percent of the experienced demand) was in force, this was indicated, along with the number of periods remaining for this penalty. Retailers could return products, up to a limit of 6 percent of cumulative shipments received. Because of the advertising lag, we did not begin demand until period 5.

(6) The earnings summary screen evaluated the retailer's performance for that period and for the entire session, using the appropriate metric.

Our markets had seven retailers of various types. Sessions lasted about three hours. Each person was seated at a computer in a carrel separated from others by dividers so that participants could not observe others' decisions.

The retailers were very different (Table 1). For example, a club retailer (Number 6) doesn't advertise,

while an Internet retailer (Number 7) has a small potential market share, keeps no stock, and has only one performance metric-revenue. We used various rates for converting participants' experimental earnings into the actual dollars we paid them, reflecting the heterogeneity of types of retailers. We also differentiated products with respect to cost and levels of demand. This experimental setup was used to test several penalty strategies (Table 2).

In the February experiments, we lengthened the sessions to seven hours, including a two-hour training period. We made a number of design changes; for example, we restricted the number of products that retailers could advertise in a period, we included a factor for historical price reputation, and we slightly modified the performance measures for the retailers. We had found that the penalties for MAP violations were ineffective near the end of the experiments (or life cycles) in the September sessions, and so we made them partially retroactive in February (Table 3).

In the training segment of each session, we presented an overview of the experiment. We summarized the mechanics involved in making choices and the effects of these choices on retailer performance. We also discussed stocking issues, service levels, pricing, advertising and demand, advertising funds, and product life cycles. With respect to product life cycles, we told participants that we would replace two products during the session and that we would notify them of this change five periods in advance. We also described the evaluation methods in some detail. We gave each participant a chart that illustrated the sensitivity of his or her own demand to advertising expenditures.

We also covered the MAP violation penalties, providing retailers with a chart of the penalties for each product in that session. Possible penalties in the September sessions included pulling products (preventing a retailer from receiving further shipments), suspending advertising funds for a number of periods, and withdrawing advertising funds for the current period. In one session, we linked all HP products, so that a violation on one product triggered penalties on all. In the February sessions, we based some penalties on net shipment value and revenue.

After a question-and-answer period in each session, participants played some practice rounds to further familiarize themselves with the mechanics involved in the experiment. We answered individual questions during this practice phase as well, and then we proceeded with the experiment (Table 4).

The simulation determined demand after the first three decisions, and retailers chose their returns after observing this demand. Each retailer made these four decisions for each of eight products in each of seven to 11 periods: how many units to order, how much to spend on advertising, what prices to charge, and how many units to return to the manufacturer.

Results

In the September sessions, we used penalties that primarily applied to future periods. We varied these penalties for products 1 through 4, the control products. We kept penalties for the remaining products constant across treatments; for products 5 and 8, a violation meant losing four periods of ad funds, while for products 6 and 7, a violation meant being fined the current period's ad expense.

Because we observed that forward-looking penalties became less effective as we neared the end of each September session, for the February sessions we made the penalties also retroactive for some number of periods. Again, we varied the penalties for products 1 through 4 and held the penalties constant for products 5 through 8. In one session, we imposed multiperiod penalties on MAP violations for products 1 through 4. In two other sessions, we removed price restrictions for either products 1 through 4 or for only products 1 and 4 (product 1 [or product 4, its life-cycle replacement] has the largest market share). We ran 20 periods in one session, eight in a second session, and 12 in a third session.

The September sessions (Table 5) differed with respect to the penalty for a MAP violation for products 1 through 4 with the violation penalty for products 5 through 8 kept constant across sessions. In the February sessions (Table 6), we imposed the restriction that a retailer could advertise at most two products in any one period.

Before moving to our analysis, we caution against imputing statistical significance to our results because

Product	September (1) Penalty	September (2) and (3) Penalty	September (4) Penalty
1	Pulled	4 periods ad funds	12 periods ad funds
2	Pulled	4 periods ad funds	12 periods ad funds
3	Pulled	4 periods ad funds	12 periods ad funds
4	Pulled	4 periods ad funds	12 periods ad funds
5	4 periods ad funds	4 periods ad funds	4 periods ad funds
6	Current period ad expense	Current period ad expense	Current period ad expense
7	Current period ad expense	Current period ad expense	Current period ad expense
8	4 periods ad funds	4 periods ad funds	4 periods ad funds

Table 2: There were four types of treatments in our September experiments. They differed in the MAP violation penalties for products 1 through 4, which represent Hewlett-Packard products. (2) and (3) also differed in the number of products to which the penalties applied. In (3), if the advertised-price restriction is violated for any of products 1 through 4, the MAP violation penalty applies to all of these products.

of the interdependence of the observations in each session. Individual sessions varied considerably, further weakening statistical comparisons. Nevertheless, we see some patterns in the data.

The overall market share of the control products was only slightly reduced by having less severe MAP violation penalties for these products. In both sets of sessions, a comparison of the harsher penalties with aggregated gentler penalties shows that the control-product market share is about 10 percent higher with the more severe penalties. While this small difference may seem surprising, it may be the result of a correlation between the pricing of the control products and the other products in any one session. Thus, HP does better with harsher penalties but only slightly.

In both September and February, we found that retailer margins were higher with the more severe penalty. This was true for both sets of products even though we held the penalties for the noncontrol products constant across treatments. This finding suggests that the retailers' pricing decisions for all goods are sensitive to the nature of the penalties for violating MAP on the control products.

In September, the average margins were about 20 percent higher when a violation led to products being permanently pulled from the retailer (for reference, we set the price restrictions so that the average margin at

Product	February (1) Penalty	February (2) Penalty	February (3) Penalty
1	3% + 3%*	No penalty	No penalty
2	3% + 3%	No penalty	3% + 3%
3	3% + 3%	No penalty	3% + 3%
4	3% + 3%	No penalty	No penalty
5	4 periods ad funds, starting this period	4 periods ad funds, starting this period	4 periods ad funds, starting this period
6	Current period ad expense	Current period ad expense	Current period ad expense
7	Current period ad expense	Current period ad expense	Current period ad expense
8	4 periods ad funds, starting this period	4 periods ad funds, starting this period	4 periods ad funds, starting this period

Table 3: There were three types of treatments in our February experiments. They differed in the MAP violation penalties for products 1 through 4, which represent Hewlett-Packard products. 3% + 3% means lose three percent of net shipment value for the past four periods plus three percent of revenue for the current period and the next three periods.

Choose	Observe	The action affects
Number of units to order	Approximate number of customers Current inventory position Buying prices (past service-level) Past selling prices	Stock available later Service levels
Advertising	Advertising budget offered Approximate number of customers Current inventory position Buying prices (past service-level) Past selling prices	Later demand
Selling prices	Approximate number of customers Current inventory position Competitors' last period prices	Prices charged customers Current period demand
Number of units to return	Demand Stock remaining	Stocking levels Service levels

Table 4: The participants made several decisions and observations in the course of the experiment. These variables were summarized and printed on reference sheets, which were handed out to participants during the experiments.

	Control products (1–4)		Other products (5–8)	
MAP violation penalty (products 1–4)	Average margin	HP share of market	Average margin	HP share of market
Lose 4 periods ad funds	0.10 (0.02)	55%	0.16 (0.03)	45%
Lose 4 periods ad funds (linked)	0.11 (0.04)	55%	0.16 (0.03)	45%
Lose 12 periods ad funds	0.08 (0.04)	41%	0.11 (0.03)	59%
Aggregated ad funds penalties	0.10 (0.03)	50%	0.14 (0.04)	50%
Pull the product	0.12 (0.01)			54%

Table 5: In the September experiments, the pull-the-product penalty was the most effective with the highest margin observed compared to the ad-funds penalties.

	Control products (1–4)		Other products (5–8)	
MAP violation penalty (products 1–4)	Average margin	HP share of market	Average margin	HP share of market
No MAP, products 1–4	0.03 (0.01)	56%	0.06 (0.03)	44%
No MAP, products 1&4	0.00 (0.09)	49%	0.04 (0.10)	51%
Aggregated no MAP	0.02 (0.06)	53%	0.05 (0.07)	47%
Backward/forward penalty (3% + 3%)	0.11 (0.04)	59%	0.14 (0.05)	41%

Table 6: In the February experiments, we found the new 3% + 3% penalty to be as effective as the previously tested (September experiments) penalties. It also maintained substantially higher margins and market share compared to scenarios to which it was not applied.

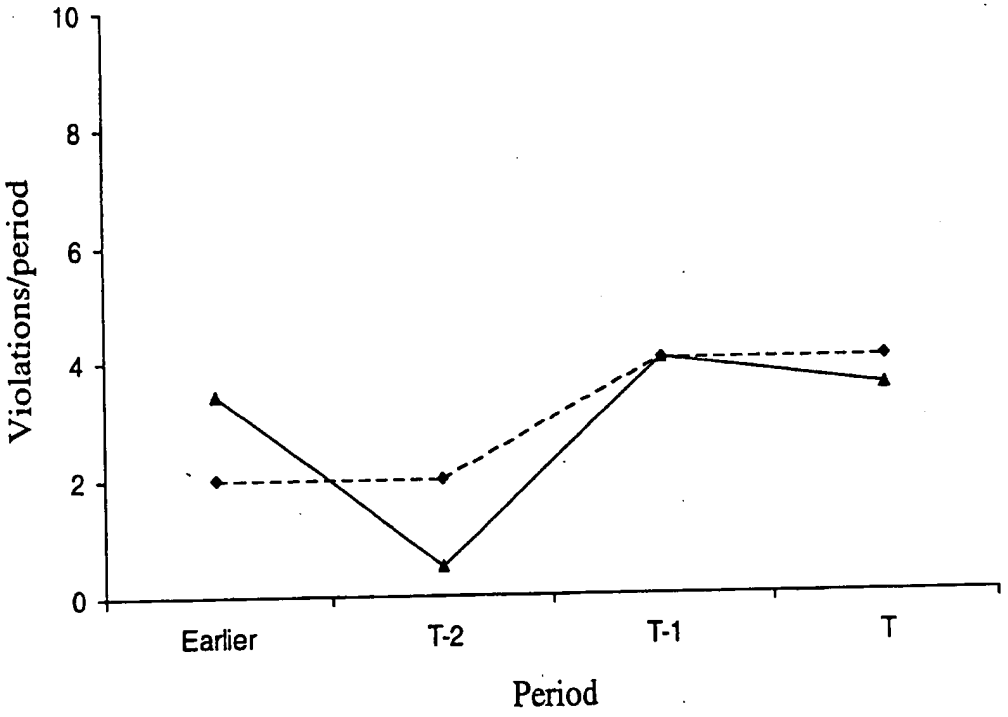


Figure 6: The MAP violations per period in the February experiments no longer showed an upward trend under either the pull-products penalty (dashed line) or the ad-funds penalties (solid line). Here we see no real time trend. This approach seems to have been effective in reducing the violation rate near the end of a session or life cycle.

escape penalties even if they violate MAP at the end of a product's life.

Discussion

Our aim in this research was to examine the effect of various penalties for violating MAPs on retailer behavior and on HP's market share. Retailer margins appear to be inversely related to the severity of the penalties for violating MAPs. Changing the penalties for the control products seemed to have only modest effects on their market share. We learned that a penalty that links products has a serious flaw, and HP decided not to use such penalties. We also found that purely forward-looking penalties led to a pattern of increasing violations as products approached the ends of their life cycles and that including a retroactive component in the penalties seemed to be effective in reducing or eliminating this effect. HP has subsequently developed a new design based on these results, introducing backward-looking penalties to counter the life-cycle effect and eliminating linked-product penalties.

Our study has many limitations, and our methodology is still evolving. We learned some lessons that might be useful to others who wish to apply experimental methods in industrial applications. A firm may wish to match its business environment as closely as possible in an experiment, but doing so may require a design that is too complex for conventional analysis. In practice, the researcher and the industrial client may need to negotiate the details of the experiment.

In addition, complex experiments may take a long time to run; we found that recruiting participants for our longer sessions was difficult. We recommend that prospective experimenters keep recruiting issues in mind. Streamlining the decision path should be helpful. We are working on new interface functionality, which should reduce the time needed for a session and

may ameliorate the recruiting problem. Our design can accommodate a variety of retailer types, an important factor given the changing markets for technology products.

The experimental approach seems promising for business enterprises wishing to evaluate the effects of policy changes, even in complex market environments. Sometimes even a limited examination of potential strategies is useful and can produce surprising dividends. Our associates in the Hewlett-Packard product divisions recognized the value of our experimental results for making business decisions and setting policies.

Acknowledgments

We acknowledge the valuable support provided by Kemal Guler, Shailendra Jain, Fereydoon Safai, and Jerry Shan (in particular, Jerry's work on the demand model) at HP Labs, and Craig Artherholt, Jacky Churchill, Richard Deep, Alex Espalin, Alan Maybruck, Sheila McKay, and Robin McShane at HP business divisions. We also gratefully acknowledge the helpful and extensive comments of two anonymous referees.

References

Brewer, P., C. Plott. 2002. A decentralized, smart market solution to a class of back-haul transportation problems: Concept and experimental test beds. *Interfaces* 32(5) 13–36.

Cason, T. 1995. An experimental investigation of the seller incentives in EPA's emission trading auction. *Amer. Econom. Rev.* 85(4) 905–922.

——, C. Plott. 1996. EPA's new emissions trading mechanism: A laboratory evaluation. *J. Environ. Econom. Management* 30(2) 133–160.

Chamberlin, E. 1948. An experimental imperfect market. *J. Political Econom.* 56(2) 95–108.

Dubin, J. 1998. *Studies in Consumer Demand—Econometric Methods Applied to Market Data.* Kluwer Academic, Boston, MA.

McCabe, K., S. Rassenti, V. Smith. 1990. Designing 'smart' computer-assisted markets. V. Smith, ed. *Papers in Experimental Economics.* Cambridge University Press, New York, 678–702.

——, ——, ——. 1991. Experimental research on deregulated markets for natural gas pipeline and electric power transmission networks. *Res. Law Econom.* 13 161–189.

McFadden, D. 1976. Quantal choice analysis: A survey. *Ann. Econom. Social Measurement* 5 363–390.

Plott, C. 1997. Laboratory experimental testbeds: Application to the PCS auction. *J. Econom. Management Strategy* 6(3) 605–638.

——. 1999. Policy and the use of experimental methodology in economics. L. Luini, ed. *Uncertain Decisions Bridging Theory and Experiments.* Kluwer Academic Publishers, Boston, MA, 293–315.

Rassenti, S., V. Smith, K. McCabe. 1994. Designing a real time computer-assisted auction for natural gas resources. W. Cooper, A. Whinston, eds. *New Directions in Computational Economics.* Kluwer Academic Publishers, Boston, MA, 41–54.

Smith, V. 1962. An experimental study of competitive market behavior. *J. Political Econom.* 70(2) 111–137.

——. 1964. The effect of market organization on competitive equilibrium. *Quart. J. Econom.* 78(2) 181–201.

Fereydoon Safai, Project Manager, Decision Technology Department, Software Technology Laboratory, Hewlett-Packard Labs, PO Box 10301, Palo Alto, CA 94303-0890, writes: "The work described in the paper has been transferred to and adopted by the Consumer Product Organization of Hewlett-Packard. Major policy decisions were made based on the information provided by Dr. Chen's and Dr. Charness's research. Jacky Churchill, Vice President and General Manager at the time of the research, wrote in an internal memo, 'The beauty of the model is its ability to allow us to "test" a number of different variations, and see the effects, without creating a "disturbance" in the marketplace.' In the past year or so, this research has also expanded to cover other policy areas and has provided substantial value to the Consumer Product Organization on many occasions."

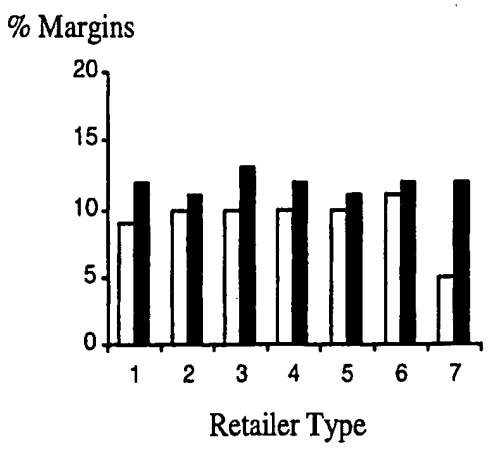


Figure 1: The September experiments showed that retailer margins for the control products are higher with the pull-product penalty (black) than with the ad-funds penalties (white).

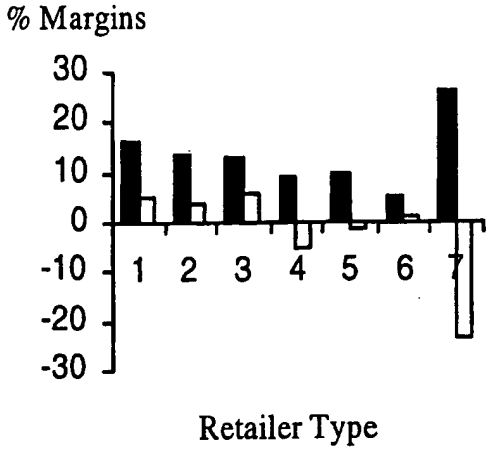


Figure 3: The February experiments showed that retailer margins are substantially higher for the control products with MAP penalties (black) than without MAP penalties (white).

the restricted price was 10 to 13 percent for the control products (Figure 1) and 17 to 20 percent for the other products (Figure 2)). If we were to assume the independence of each observation, this difference would be statistically significant at $p = 0.04$ (one-tailed test).

The margins were always lower for retailers when the penalty for violating a control-product MAP was only temporary. Even though the penalties vary only for the control products, this is true for all 14 comparisons.

The difference in margins was even more pronounced in the February sessions (Figures 3 and 4). The margin with MAP is significantly higher than the margin for the combined sessions without MAP at $p = 0.002$ (one-tailed test).

It is apparent that the margin for individual retailers on all products is robustly higher with strict penalties for MAP violations.

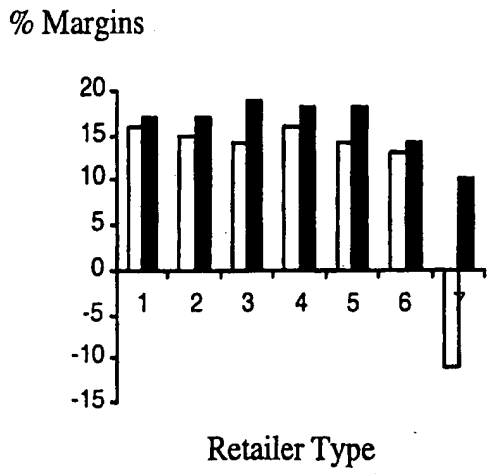


Figure 2: The September experiments showed that retailer margins for other products are higher with the pull-product penalty (black) than with the ad-funds penalties (white).

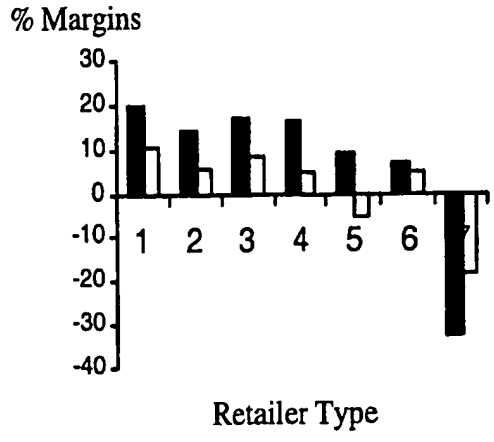


Figure 4: The February experiments showed that retailer margins are substantially higher for the other products with MAP penalties (black) than without MAP penalties (white).

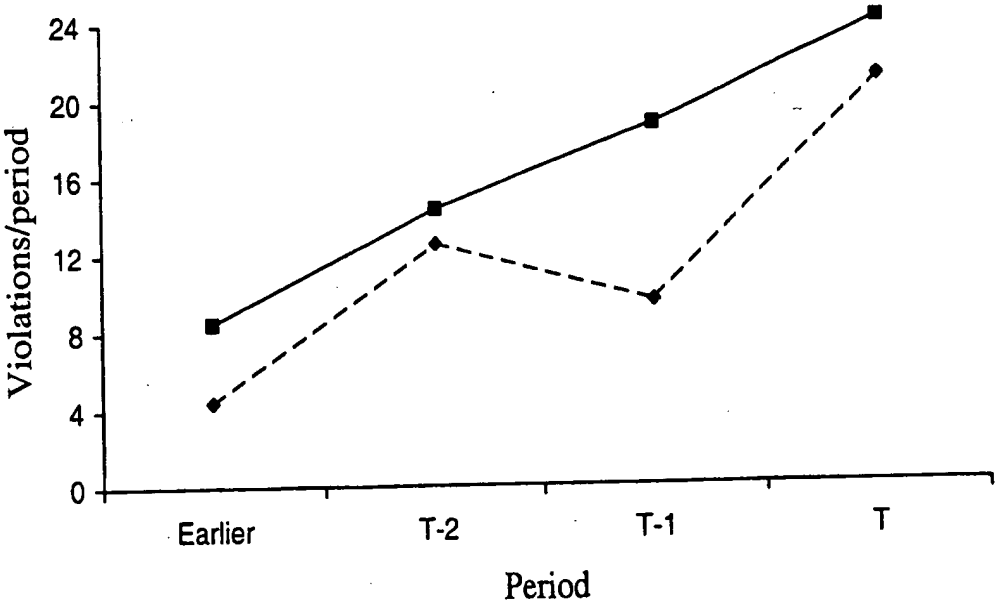


Figure 5: MAP violations per period in the September experiments showed an upward trend under both the pull-products penalty (dashed line) and the ad-funds penalties (solid line). T represents the final period of a session, T-1 the penultimate period, and so forth.

In the September sessions, we used an exclusively forward-looking violation. We observed a pattern in the violation rate over time: close to the end of the experiment, every retailer violates MAP substantially more. A forward-looking penalty should (and did) have diminishing effectiveness as a product approaches the end of its life cycle (Figure 5).

We see a positive time trend in the number of violations per period, as there are more violations as the end of the life cycle approaches. In the February sessions, we introduced a violation penalty with a retroactive component.

We found that the frequency of violations was related to the form of MAP imposed. We also found that retailers (particularly mid-sized retailers) did not fare as well without MAP, as their margins were distinctly smaller; interestingly, removing the MAP on some products affects the margins for both those products and for the others. This calibration suggests that equilibrium prices may well be below the price floor. Based on our results, HP felt it would be best to continue some form of MAP.

We were also able to detect weaknesses in the design and enforcement of several advertised-price policies; this led HP to revise the policies it implemented. For example, retailers may carry several different HP products. One proposed enforcement policy would link these products, so that a violation on any individual product would trigger penalties on all of them. When we tested this policy, we found that retailers who decided to violate the MAP on one product would often violate the MAP on all the linked products. As a result, HP decided not to implement a linked-product MAP design.

In addition, in our first set of sessions, we identified a problem with respect to MAP and product life cycle. Initially, we tied MAP penalties to future shipments and future market-development funds for the product at issue. However, we found that the violation rate increased toward the end of the life of a product. Retailers correctly perceived that forward-looking penalties would have little effect late in a product's life. Because of this, HP decided to adopt a completely different enforcement policy, which we validated in our second set of sessions (Figure 6). This new policy is retroactive as well as forward looking, so that retailers cannot